## *Mobile Application Development (CPRG-303-E)*

## Project Phase 2: App Architecture

## **Prepared by**

***Coffee Mates***

*Team Name*

Members:

## *Claur, Alessandra Nicole*

## *Escartin, Mitzi Vera*

## *Fontelo, Lulubelle*

## *Siwa, Gabriel Ira*

*Teruel, Bryan Benedict*

**Prepared for**

Mohammed Al-Haifi

Date: March 05, 2025

## 

## 

## 

## **School for Advanced Digital Technology**

## **1. Introduction**

This document details the architectural decisions for Studdy, a mobile study planner designed to help students manage their tasks, set reminders, and track progress efficiently. This architecture plan is based on the team's skill set and project requirements, ensuring a structured development process.

## **2. Architectural Framework Overview**

## The Studdy mobile application follows a three-tier architecture, ensuring modularity, scalability, and ease of maintenance:

## **2.1. Architecture Diagram**

A diagram of a study planner

AI-generated content may be incorrect.

## **2.2 Presentation Layer (Frontend)**

## Built using React Native for cross-platform support.

## Utilizes Bootstrap for UI consistency.

## Implements React Navigation for screen transitions.

## UI components include Task List, Calendar View, Reminders, and Progress Tracker.

## **2.3 Business Logic Layer**

## Handles task management, scheduling reminders, and progress tracking.

## Ensures seamless navigation between screens using stack-based routing.

## **2.4 Data Layer (Storage & Backend)**

## Local Storage: Encrypted AsyncStorage for offline access.

## Cloud Storage: Firebase for cloud synchronization and multi-device access.

## Data Flow: UI interactions → Business Logic Processing → Data Storage & Retrieval.

## 

## **3. Architecture Decision Records**

### **3.1 Development Framework**

# ADR-001: Development Framework

## Status <Pending> - Everyone in the team needs to confirm that they agree to this framework

## Context Our mobile app is a \*\*Study Planner App\*\* that allows students to create tasks, set reminders, and receive notifications. We need a framework that supports cross-platform mobile app development with Android as the target device.

## Decision

We will use \*\*React Native\*\* as our development framework.

## Justification

- It supports Android devices.

- The team is familiar with JavaScript and React.

- It provides a wide range of libraries for navigation and notifications.

## Consequences

- The app will require third-party libraries like `react-navigation` and `react-native-push-notifications`.

- Some advanced features may require native code integration.

### **3.2 Navigation Strategy**

# ADR-002: Navigation Strategy

## Status Accepted (During 02/28 meeting)

## Context

The app needs a simple way for users to navigate between the \*\*Home\*\*, \*\*Calendar View\*\*, and \*\*Settings\*\* screens.

## Decision We will implement

\*\*Bottom Tab Navigation\*\* using the `react-navigation` library with the following structure:

- Home Tab: Task List and Daily/Weekly Overview

- Calendar Tab: Calendar View

- Settings Tab: App settings

## ## Justification

Bottom Tab Navigation provides a user-friendly and intuitive way to switch between primary app sections. It aligns with standard mobile design patterns, ensuring a familiar experience for users. Additionally, it keeps navigation accessible from all screens, reducing the number of steps needed to move between key features.

## Consequences

- The app will have a consistent user experience across all screens.

- We need to install and configure `react-navigation` dependencies.

### **3.3 Hardware Utilization**

# ADR-003: Hardware Integration

## Status Accepted

## Context

Our app requires hardware integration to improve user experience and productivity.

## Decision

We will integrate the following hardware features:

- \*\*Push Notifications\*\*: Remind users of upcoming tasks.

- \*\*Vibration\*\*: Provide haptic feedback for notifications.

We will not integrate GPS, Camera, or Fingerprint Scanner, as they are not relevant to the app's functionality.

## Consequences

- The app will request user permissions for Notifications access.

- Vibration feedback will be limited to supported devices.

### **3.4 Database Storage**

# ADR-004: Database Storage

## Status

<To be confirmed - wala pa usapan about storage>

## Context

The app needs to store user tasks, reminders, and completed task history.

## Decision

We will use \*\*Local Storage (Encrypted)\*\* with `AsyncStorage` in combination with \*\*SQLite\*\* to:

- Store tasks locally on the device.

- Encrypt sensitive user data.

- Sync with the device's calendar for recurring events.

## Consequences

- The app will work offline without needing an internet connection.

- Data will be stored securely, but syncing with external services will require additional development in the future.

## **4. Progress Report**

## **5. Attribution List**

|  |  |
| --- | --- |
| **Task** | **Assigned Member** |
| Wireframe |  |
| UX/UI Design |  |
| Navigation and Notification Integration |  |
| Documentation and Report Compilation |  |
| Research and Decision Validation |  |